NT DII COE

Application Installation

Proposal

27 October 1998

Presented by
Andrew Cox
PMW 157-1

Current NT Options

- Full segmentation
  - » Package install executable using MakeInstall
- Partial (COTS)
  - » Use COTS installer package to install, and use COEInstaller to subsequently install IRT&S data
- Rely on commercial rules to provide the knowledge/discipline for application environment
  - » Make all application developers submit products to the Veritest to determine “compliancy”

Problems/Assumptions

- Installshield by itself is too flexible
  - » Developers can package segments with as much (or as little) information as they feel
- Logo certification (by itself) is inadequate
  - » Only determines how well Win32 apps behave
  - » Does not handle a “federation” of software
    - – Conflicts
    - – Dependencies
    - – GOTS shared services not part of *\windows\system32*
- The fleet tends to “upgrade” installations
  - » Removing files and directories that are not associated with known apps/operating system configuration (e.g., SegDescrip)
  - » Install unapproved software using commercial tools

Problems/Assumptions

- Current NT DII COE installer does not support native NT features
  - » Difficult to account for changes in NT installation features (NT 5 and Active Directory, MSI)
  - » Developers must package apps twice to meet COE compliance AND insure use of Registry and compatibility with other native apps
- Installing applications must be able to “see” new applications/files/libraries for rogue applications or configurations
  - » maximum protection against conflicts and dependencies
  - » operate when native de-install mechanism is used

Objectives

- Create a Standardized Application Installation and Validation process for use by NT Application Developers.
- Maximize use of COTS technologies
  - » provide a path to NT 5.0
    - – Active Directory
    - – Microsoft Installer
    - – Zero Administration Workstation
- Adhere to NT Logo installation Requirements
  - » Automate the Creation of Installation Package
  - » Provide easy interface to input Logo requirements
- Innovative Approach to satisfying I&RTS Tenants
  - » Try to meet the IRT&S requirements while making maximum use of what native NT environment offers

NT Logo Compliance Installation Rqmts

- Be Easy to Install and Remove
- Provide Graphical 32-bit setup
- Provide for Unattended Installation
- Support Installation from multiple CD-ROMs
- Support Add/Remove Programs
- Avoid Rebooting the System
- Detect and Handle Versions
- Register an Uninstaller

NT Logo Compliance Uninstallation Rqmts

- Remove All Application Files during Uninstall
- Remove all References from the Start Menu
- Remove Registry Entries during Uninstall
- Remove Uninstaller as the Last Step of Uninstall

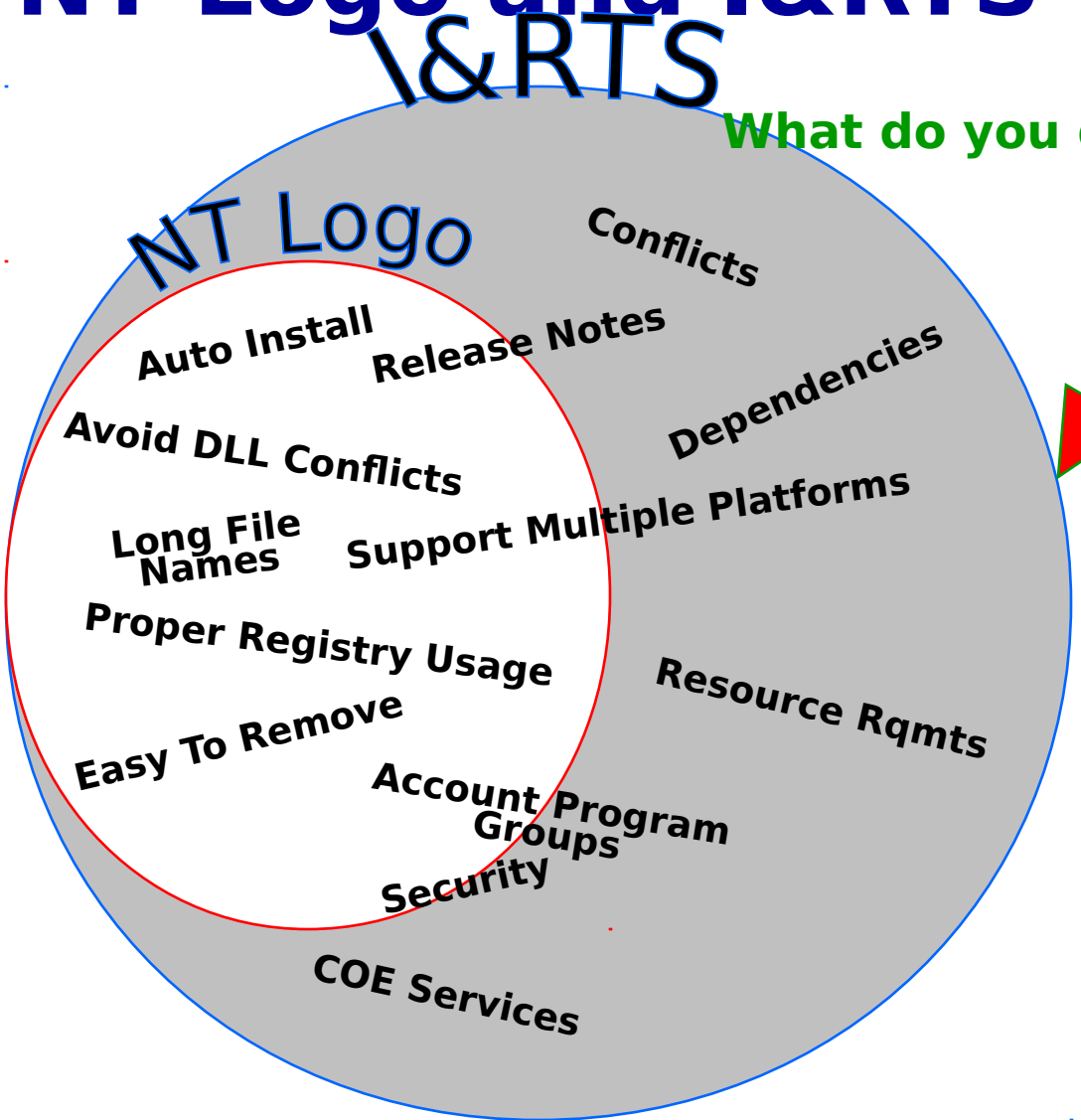
Initial Recommendation InstallShield 5.1

- Supports NT Logo Compliant Installation
  - » Writing NT 5.0 Windows Installer
- Support for Install from the Web
- Programmable (C like scripting language)
- Allows Execution of 3rd party applications within InstallShield
- Supplies and Registers an Automated Uninstaller
- Logs all of its Actions
- Supports Interactive and Silent (hands-off) Installation
- Checks for User Privilege Level and Installs Accordingly

Recommended Application Descriptor

- Must Support the DII COE I&RTS Rqmts
  - » Conflicts, Software/File Dependencies
  - » System Requirements
  - » Shared Files, File Attributes, Permissions
  - » COE Services
  - » Version Control
- Must Support NT Logo Compliance
  - » Developer Information, Versions, Release Notes, Help
  - » Efficient Use of Registry

Relationship Between NT Logo and I&RTS



What Should Go in the Registry?

Source:
Inside Windows 95 Registry
by Ron Petrusha

- Application Settings
  - » NT Logo Compliant Information
  - » Installation and Licensing Information
- User Settings
  - » Configurable Options: Tool Bars, Default Directories, Colors, etc
- Sound Events
  - » Program Errors, Closing Windows, Warnings, Messages, etc
- Application State Information (position)
- Most Recently Used (MRU) Files List

Registry Storage Space

Source:
January 1998 MSDN

- Generally, data consisting of more than one or two kilobytes (K) should be stored as a file and referred to by using a key in the registry
- Each Self Registering File Creates 50 Registry Entries
- Limit 55,000 Entries

What Should NOT go in the Registry

- Crime
  - Saving System Info That Is Available Via Windows API
  - Saving Non-Application State Like Information
  - Exceeding Max Configured Registry Size
- Punishment
  - Performance Degradation
  - Error: Cannot Write to Registry (serious)
  - Error: Corrupt Registry (deadly, must re-install Windows)
  - Non-Compliance with Future Limitation on Registry Size

Recommended Application Descriptor File Format

- Extensible Markup Language (XML)
  - » an emerging Internet standard based on SGML
- Human Readable
  - » Personalized Cascading Style Sheets can customize view
- Supports User Defined Tags (Document Type Definition)
  - » *DTDs provide the blue print for the words (tags) and grammar that associated XML files can use.*
  - » Additional tags can be added without breaking parser
- Free Parsers Available
  - » Microsoft, Java, Netscape, LT
- Major Browser Developers Pledged Support
- Language Independent

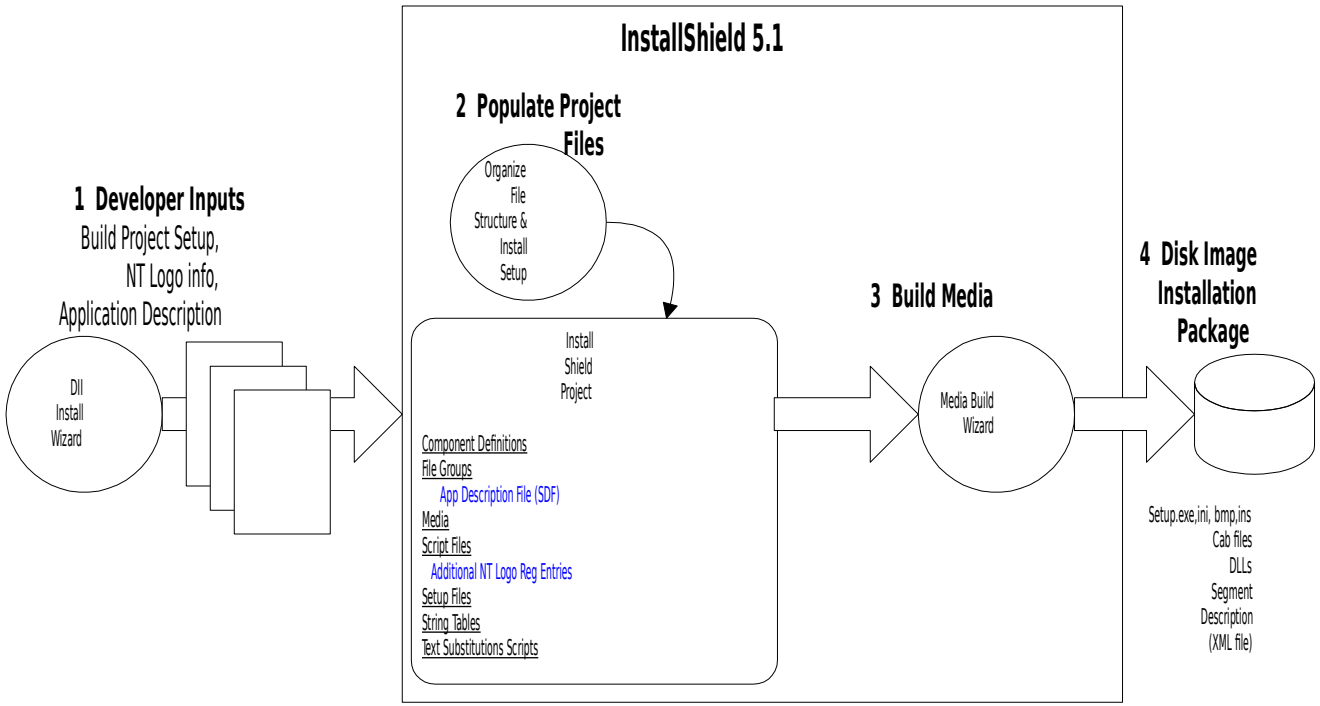
Sample Document Type Definition (DTD)

```
<!ELEMENT Name (#PCDATA)>
<!--

*************************************************************************
    ** The Name of the Segment with It's classification required.
    ** The Id element is the Unique Id of the Segment in the DOL
Database

*************************************************************************
-->
<!ATTLIST Name
    Id              ID  #REQUIRED
    Classification (UNCLASS|CONFIDENTIAL|SECRET|TOPSECRET) "UNCLASS">

<!--

*************************************************************************
    ** A name/version structure used to track
    ** required segments and conflicts

*************************************************************************
 -->
<!ELEMENT VersionNumber (#PCDATA)>
<!ELEMENT SegmentName (#PCDATA)>
<!ELEMENT SegmentVersion (SegmentName, VersionNumber)>

<!--
```

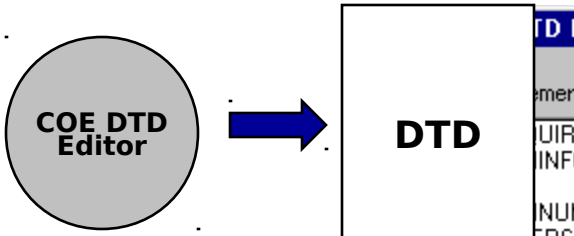
Sample Application Description XML File

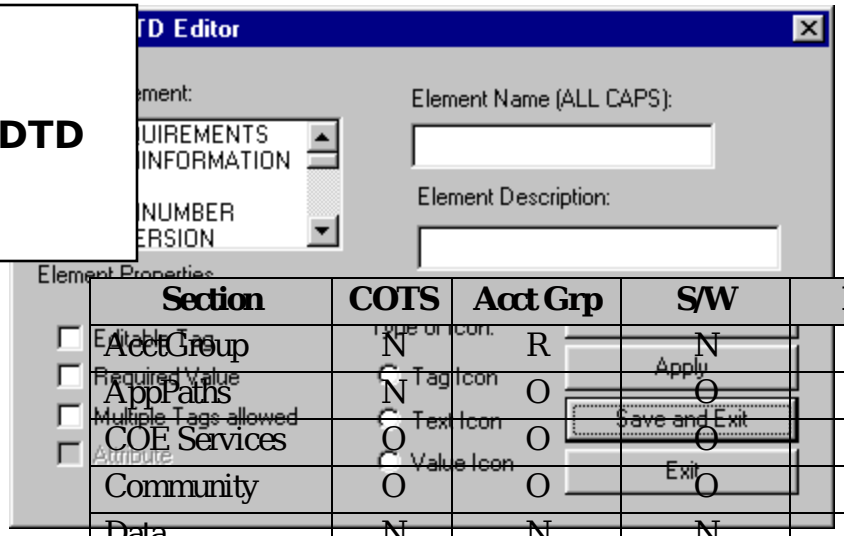
```
<?XML VERSION="1.0"?>
<!DOCTYPE SegmentDesc SYSTEM "SegmentInfo.dtd">
<SegmentDesc>

<!--

*************************************************************************
    ** The Name of the Segment with It's classification required.
    ** The Id element is the Unique Id of the Segment in the DOL
Database

*************************************************************************
-->
    <Name Id="Sample-24" Classification="UNCLASS">Sample Segment</Name>


<!--

*************************************************************************
    ** The Version number of the Segment

*************************************************************************
-->
    <VersionNumber>3,1,0,4</VersionNumber>

<!--
```

Installation Package Development Process



Step #1

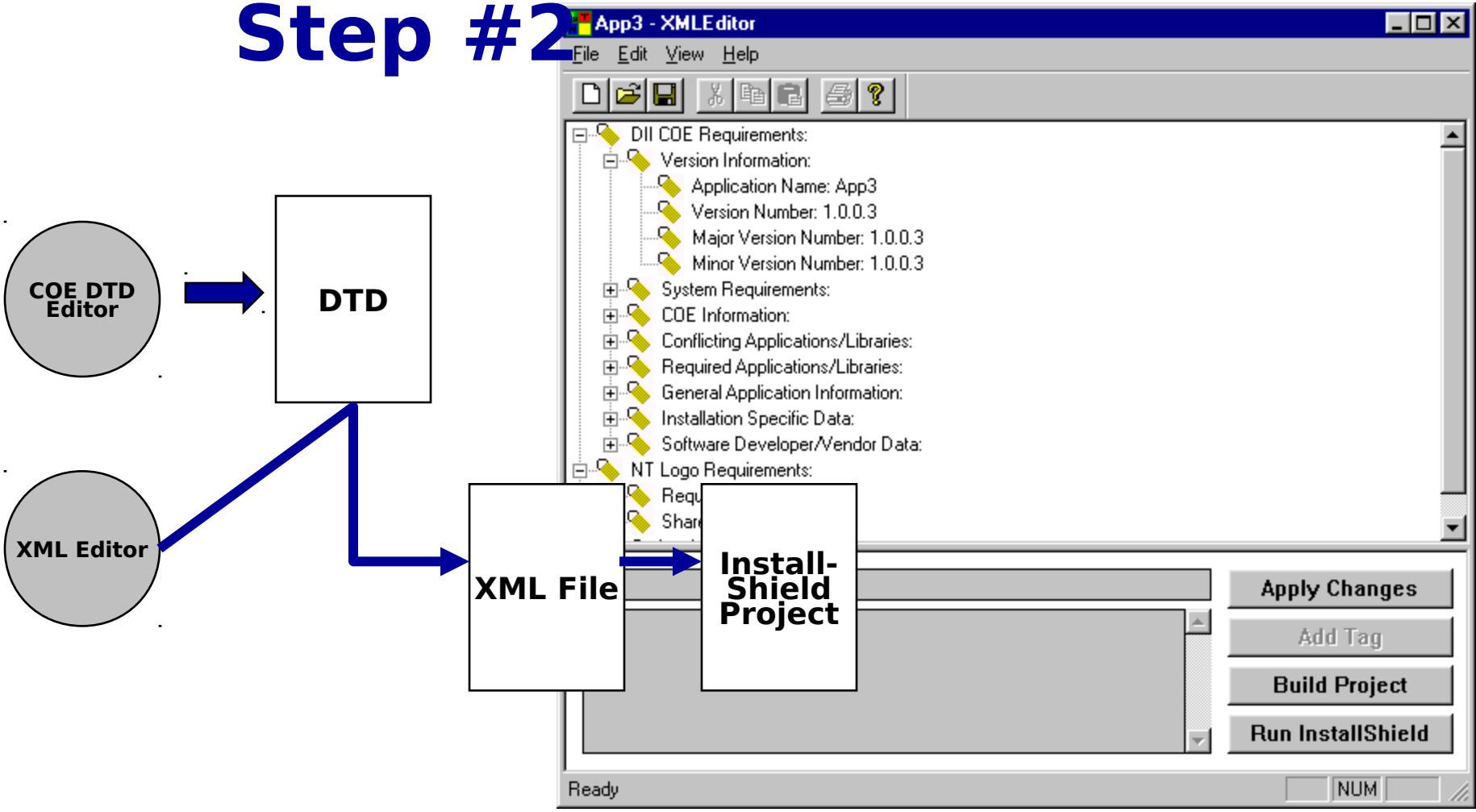




Section	COTS	Acct Grp	S/W	Data	DB	Patch
AcctGroup	N	R	N	N	N	N
AppPaths	N	O	O	N	N	N
COE Services	O	O	O	O	O	O
Community	O	O	O	O	O	O
Data	N	N	N	R	N	N
Database	N	N	O	N	R	O

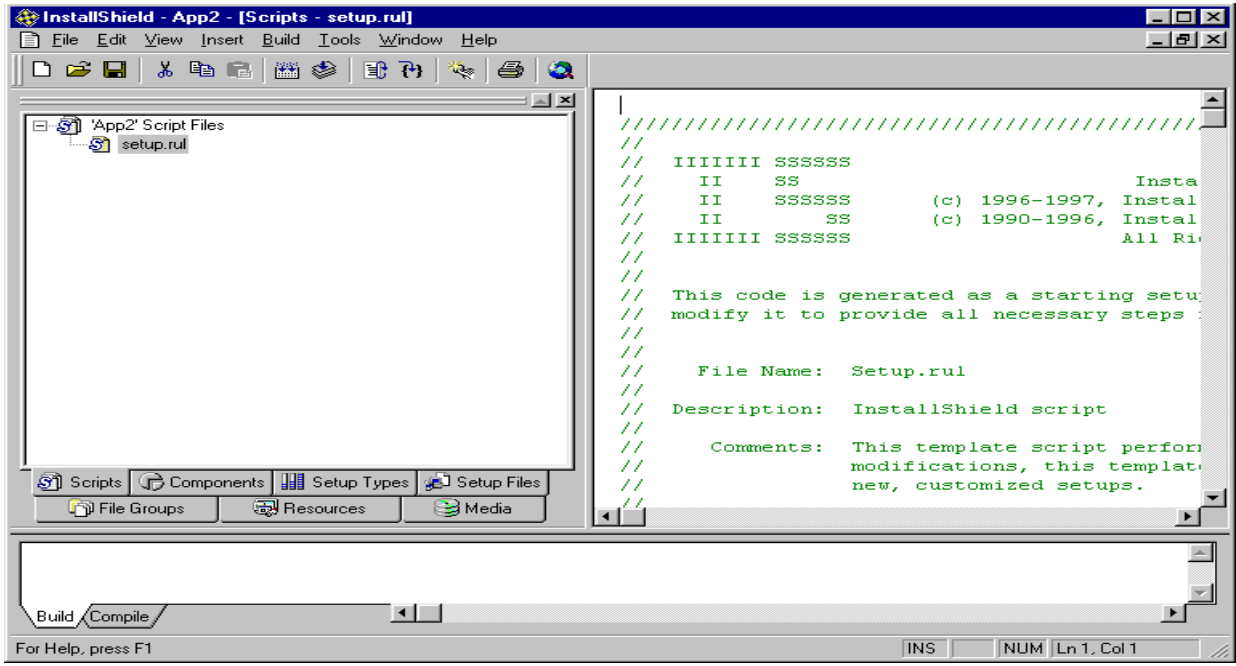
- Launch COE DTD editor to customize tag variables, set required tags, and determine parent/child relationships, tag display names
- Allows engineering offices set additional tags other than IRT&S and NT Logo requirements (and make them required) without breaking the model or requiring ANY recompile, redesign
- Need to automate the IRT&S requirement matrix

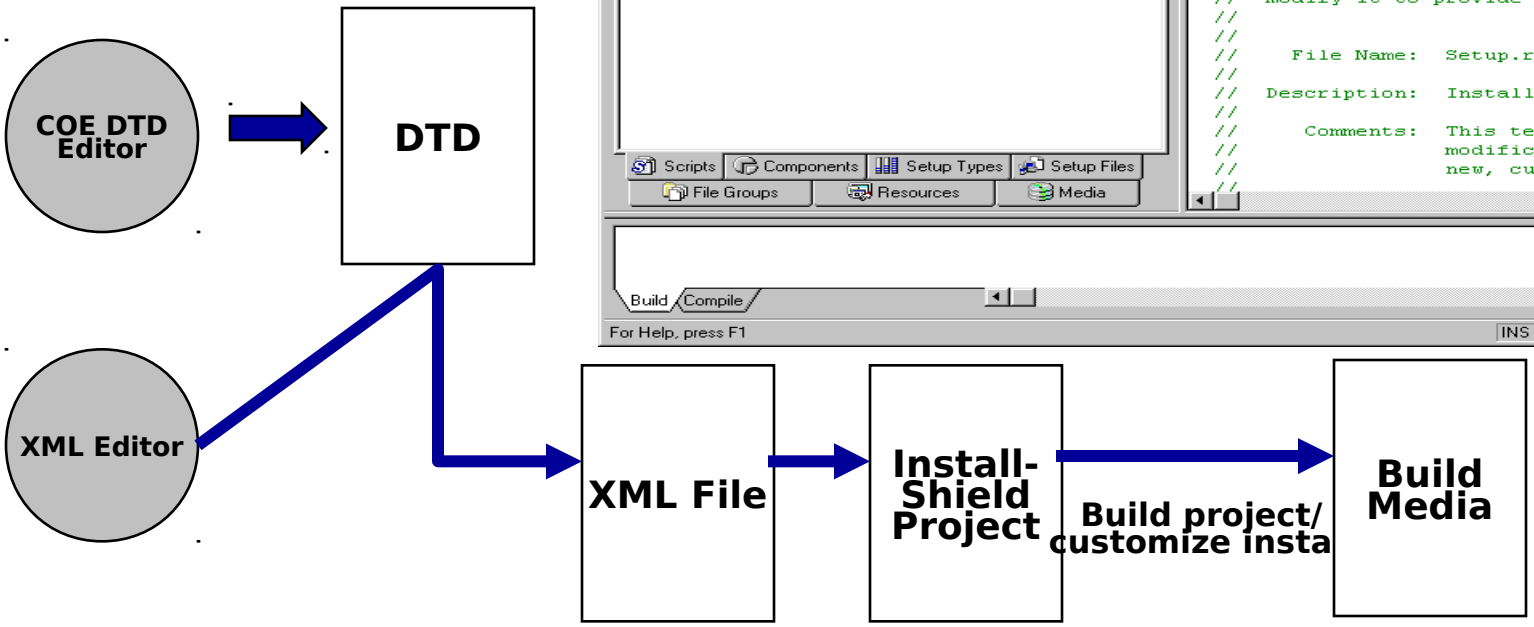
Step #2



- Launch the XML Editor to enter all data in tag entries
- Build project file, which generates XML file and InstallShield project file
  - » Cannot be done unless required entries are made
- Click Run Installshield to bring up project in InstallShield

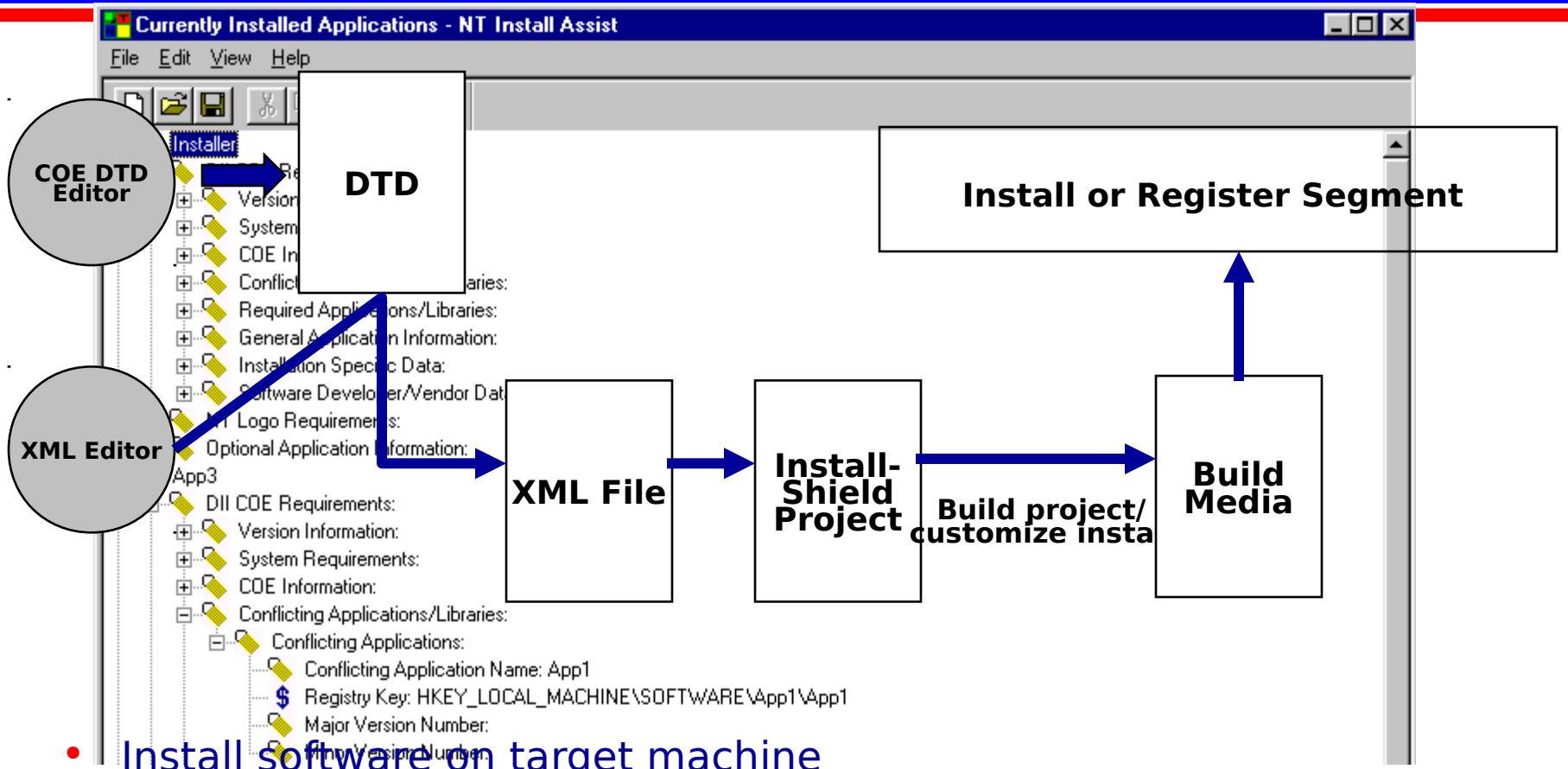
Step #3





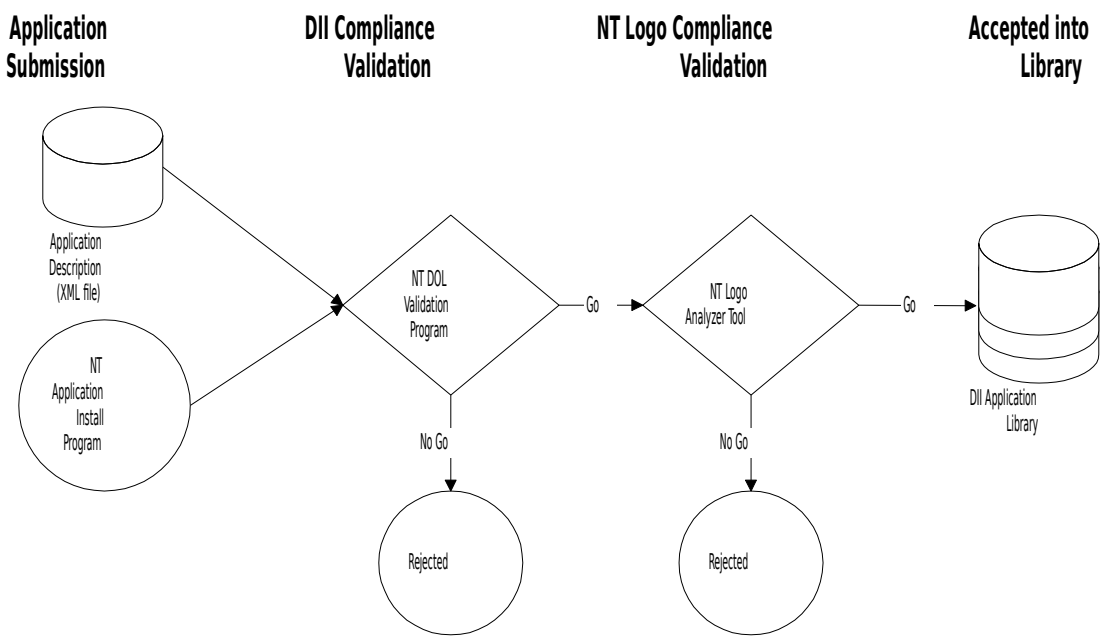
- Populate Installshield template and build executable and media
- All XML Editor values are saved into application XML file
  - » Includes all designated shared files, etc.
  - » Includes all NT Logo registry requirements/entries
  - » XML file is bundled in shared file list in installshield for install
- Use Installshield to continue customization

Step #4

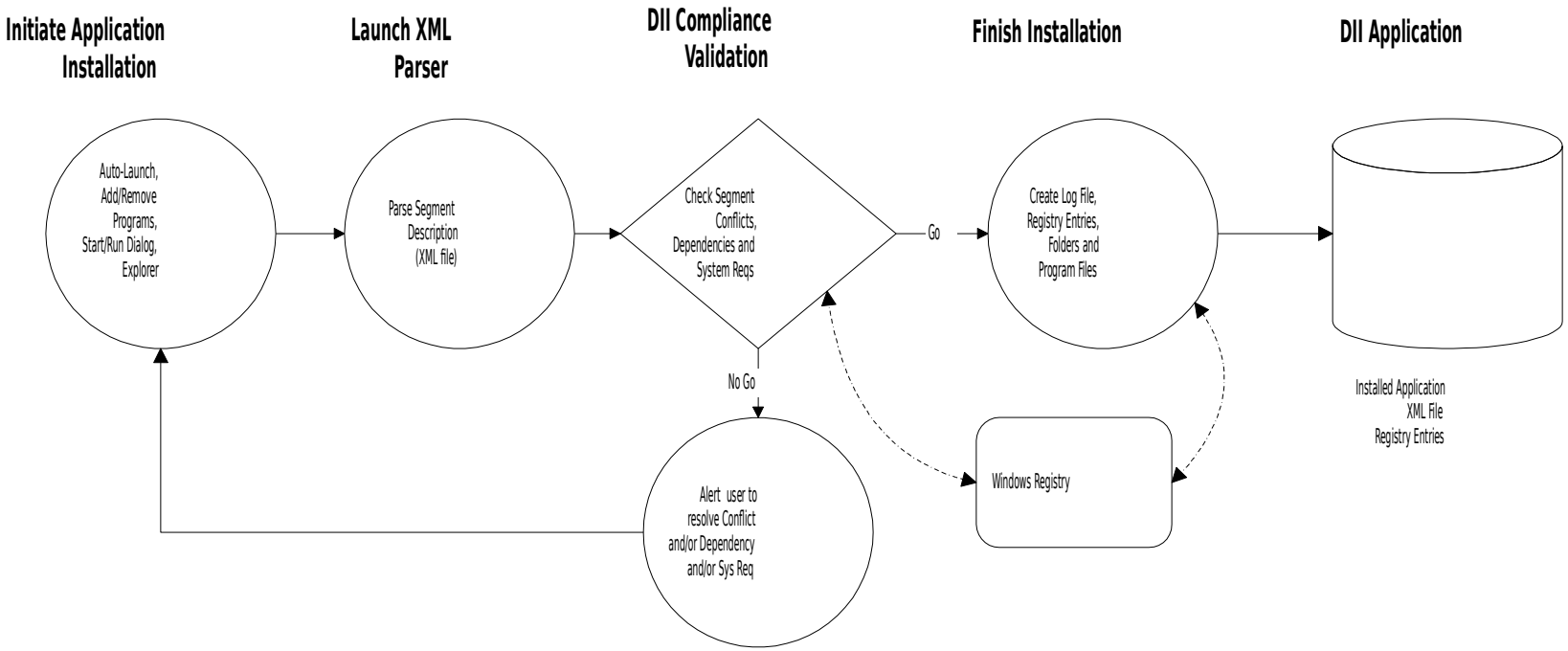


- Install software on target machine
  - » All Logo entries are made
  - » XML file is installed in application directory
- Application can be viewed with name and version # in Control Panel
- Application IRT&S data can be viewed with text/browser or COE Viewer

NT DII Application Library



Application Installation Process



What did DISA Say?

- On NT, DISA will go the way the Navy goes with the current installer
  - » still waiting for committment to DII COE 3.4 or 4.0
  - » looking for path to get to JPL
- “Hey, Randrew, go figure out the closest point of departure from UNIX segments”
  - » put together a proposal to get as close UNIX to NT parity as possible
  - » everything from here on is a “***proposal***”, and has not gotten DISA approval

DII COE Installer Proposal

- XML is platform independent
- Build an SDF to XML translator
  - » Very easy, low risk
- NT Apps that ONLY used COEInstaller
  - » Run SDF conversion utility
  - » Open XML file in XML editor and rebuild (EASY!)
- NT Apps that used COEInstaller to launch InstallShield
  - » Run SDF conversion utility
  - » Even if Logo compliant, need to rebuild (EASIER!)

OSD Proposal

- Create segment description documents using the Open Software Description (OSD) format. By extending the OSD format for our needs, the installation specific requirements of the DII COE Integration and Runtime Specification (I&RTS) can be met
  - *OSD is an open standard DTD that defines tags that are very specific to the distribution of software.*
  - *The OSD format contains a starting set of tags from which to build our XML document. It also provides a common context for storing IRT&S data so that integration with COTS packaging and delivery mechanisms (e.g., Channel Definition Format) is more seamless.*
  - *Marimba, Tivoli, and Microsoft Initiative*

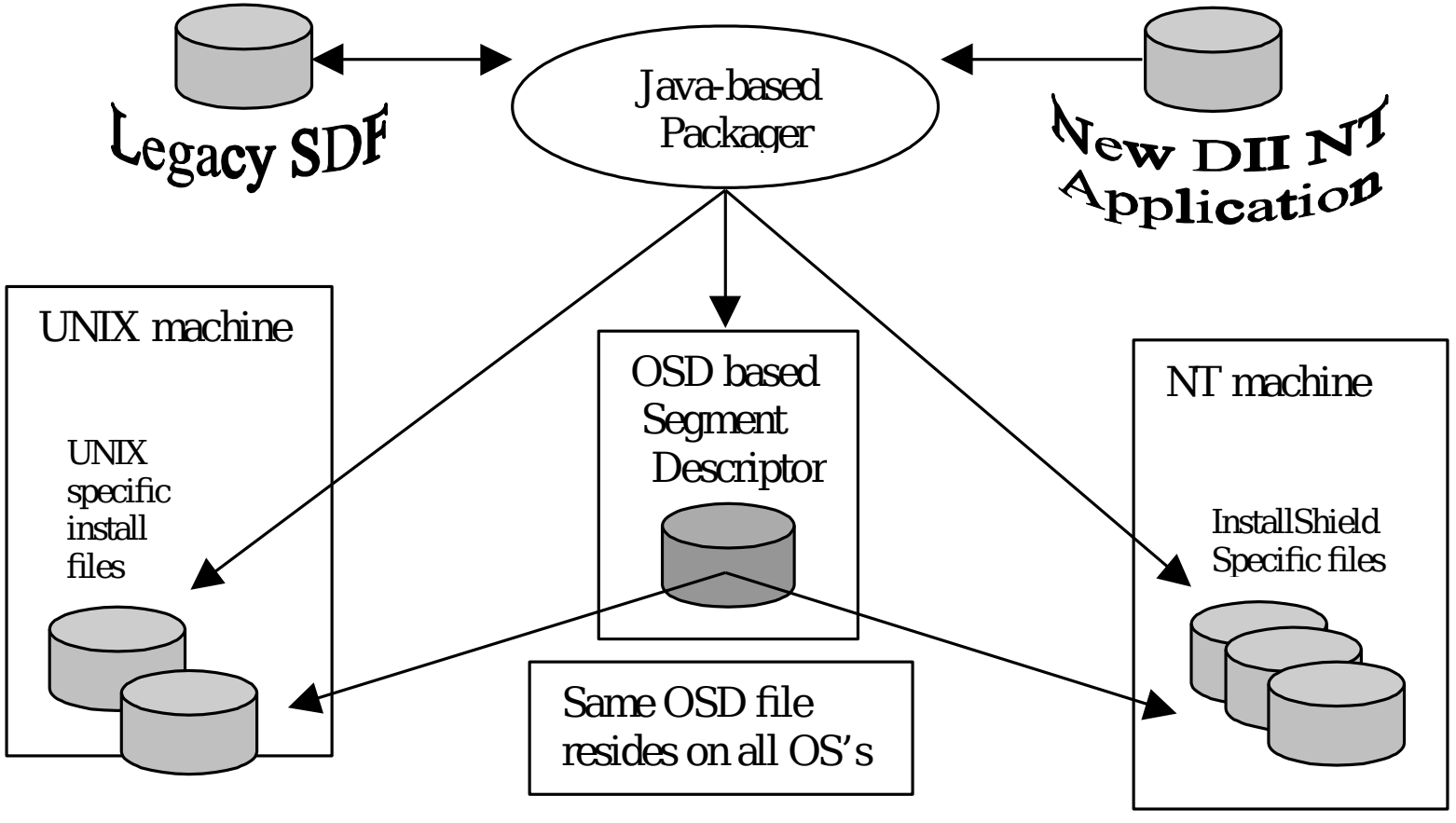
Cross-Platform Approach

- Create a Java based Packaging tool that can be used on any Operating System hosting a Java Virtual Machine.
  - » Create a segment description file in our extended OSD format.
  - » In addition, the tool will create all externally required “project” files to be used with COTS Install programs.
- Installshield has committed to providing an Java Edition on Sun (HP?)
- Sun has seen approach and is investigating applicability to other efforts (HP?)
- Gives DISA/Navy the ability to send RFP to commercial installation tools to support OSD

The Interim?

- For HP-UX, the OSD editor will create both an OSD file and the "legacy" SDF format so that the data can be built once and "auto-launch" the MakeInstall utility until a viable commercial product exists
  - » Enables developers to at least use a single application “wrapping” interface
  - » No requirement to use the old MakeInstall

The Picture



COTS installers use the OS specific files to create self-extracting archives…

OSD DTD

```
<!ELEMENT ABSTRACT (#PCDATA)>

  <!ELEMENT CODEBASE EMPTY>
  <!ATTLIST CODEBASE FILENAME CDATA #IMPLIED>
  <!ATTLIST CODEBASE HREF CDATA #REQUIRED>
  <!ATTLIST CODEBASE SIZE CDATA #IMPLIED>

  <!ELEMENT DEPENDENCY (CODEBASE|SOFTPKG)* >
  <!ATTLIST DEPENDENCY ACTION (Assert|Install) "Assert">

  <!ELEMENT DISKSIZE EMPTY>
  <!ATTLIST DISKSIZE VALUE CDATA #REQUIRED>

  <!ELEMENT IMPLEMENTATION (CODEBASE | DEPENDENCY | DISKSIZE |
    IMPLTYPE | LANGUAGE | OS | PROCESSOR | VM)*>

  <!ELEMENT IMPLTYPE EMPTY>
  <!ATTLIST IMPLTYPE VALUE CDATA #REQUIRED>

  <!ELEMENT LANGUAGE EMPTY>
  <!ATTLIST LANGUAGE VALUE CDATA #REQUIRED>

  <!ELEMENT LICENSE EMPTY>
  <!ATTLIST LICENSE HREF CDATA #REQUIRED>

  <!ELEMENT MEMSIZE EMPTY>
  <!ATTLIST MEMSIZE VALUE CDATA #REQUIRED>

  <!ELEMENT OS (OSVERSION)*>
  <!ATTLIST OS VALUE CDATA #REQUIRED>

  <!ELEMENT OSVERSION EMPTY>
  <!ATTLIST OSVERSION VALUE CDATA #REQUIRED>

  <!ELEMENT PROCESSOR EMPTY>
  <!ATTLIST PROCESSOR VALUE CDATA #REQUIRED>

  <!ELEMENT SOFTPKG (ABSTRACT | IMPLEMENTATION | DEPENDENCY | LICENSE |
    TITLE)*>
  <!ATTLIST SOFTPKG NAME CDATA #REQUIRED>
  <!ATTLIST SOFTPKG VERSION CDATA #IMPLIED>

  <!ELEMENT TITLE (#PCDATA) >

  <!ELEMENT VM EMPTY>
  <!ATTLIST VM VALUE CDATA #REQUIRED>
```